Technical Disclosure Commons

Defensive Publications Series

January 13, 2017

WEBPAGE HIDING FOR ANTI-FLICKERING SYNCHRONOUS SCRIPT LOADING

Brian Kuhn

Dimitrios Dimitropoulos

Bing Chen

James Wogulis

A/B testing, also known as split testing or bucket testing, is a term for a randomized experiment in which two variants, a control variant and an experimental variant, are tested. In an online context, A/B testing is used to identify changes to a webpage or a mobile application that increase or maximize an outcome of interest. Specific portions of a webpage, such as the webpage's paragraph text, testimonials, links, images, social media mentions, awards and badges, and calls to action text and buttons may be individually tested. The developer may make minor changes, such as changing the font color, or may completely redesign the webpage content and layout. Visitors of the website are either served the original version of the webpage or the modified version, and the A/B testing tool records and statistically analyzes the visitors' interaction with each webpage to determine whether the modified version improves the website's ability to increase web traffic and drive business goals (e.g., increase online transactions for products and services, new subscriptions, and click-through rate).

A common issue with A/B testing is a flickering effect, caused by a transient display of the original webpage before the new webpage with modifications loads and takes effect. Many A/B testing tools that use Document Object Module (DOM) modification methods provide mechanisms to prevent flickering, such as the running of customized scripts ("optimization scripts"). However, these scripts may only eliminate the occurrence of flickering once they load and execute within the webpage under test. Consequently, A/B testing tools require synchronous loading of the optimization scripts within the webpage to ensure that the flickering prevention mechanism launches prior to the display of the webpage.

Although the synchronous loading of the optimization scripts usually resolve the flicking effect, they have drawbacks making their use undesirable. For example, the webpage must fetch the synchronous scripts from a remote optimization server before loading the script. The delay in sending the request and waiting for the script to arrive adds noticeable delay to the loading of the webpage, which interferes with the visitor's website experience. In more extreme cases, the scripts may completely break the webpage. This may occur when the remote optimization server becomes unresponsive due to servicing other third-party script requests or during unpredictable network connection issues.

Webpage hiding

A/B testing tools may eliminate the flickering effect, without introducing additional delay to the loading of a webpage, by implementing a webpage hiding method together with the asynchronous loading of the A/B tool's optimization scripts. The webpage hiding method does not override an A/B tool's comprehensive flickering prevention mechanism. Rather, the webpage hiding method complements the A/B tool's flickering prevention mechanism by hiding the whole page, but only until the flickering prevention scripts fully load and execute. At that point, the flickering prevention mechanism takes effect and only hides the specific elements of the page that the web developer selected to modify, which typically have a negligible effect on the webpage's perceived responsiveness.

Asynchronous-hide snippet

To use the method, the website owner embeds a small compressed script in the form of an "asynchronous-hide snippet" into their webpage. Snippets are small, compact, and relatively easy for a computing system to execute, so website owners can be confident that they don't interfere with the webpage performance. The webpage hiding method is designed to be configurable, but simple by default. Therefore, a website owner that goes through the process of decoding it can easily appreciate that it has no side-effects other than hiding the page up to a maximum time-limit. That is, other software code may not override the snippet behavior to keep the page hidden long after the expiration of a time-limit. The key features of the snippet include:

1) A webpage-hiding method to hide the whole webpage during the loading of optimization scripts.

2) An "end-of-hiding" function that reverts the effects of the webpage-hiding method. This function is stored in a globally accessible object that asynchronous script clients can access once they load.

3) A timer that ensures that the end-of-hiding function will be always called when the time limit exceeds. It will also remove the end-of-hiding function to indicate that the timeout has occurred.

4) A globally accessible data structure used to uniquely identify asynchronous scripts or other clients that require hiding, together with their state.

endHide Function

Furthermore, the webpage hiding method includes a small "endHide" function, separate from the "end-of-hiding" function embedded in the asynchronous-hide snippet, to implement the un-hiding protocol. The webpage hiding method invokes the endHide function on start-up by any asynchronous-hide client, such as the A/B testing asynchronous scripts which are uniquely identified by an identifier string. The key operations performed by the endHide function include:

1) Ensuring that the asynchronous-hide snippet is available within the page.
2) Ensuring that the current asynchronous-hide script is a registered client in the asynchronous-hide mechanism.
3) Marking the "not-requiring-hiding" element (stored in the client state globally accessible data structure) to indicate that the state of the current script no longer requires hiding.
4) If all registered clients in the client state data structure are marked as not-requiring-hiding, then call the end-of-hiding function (this is globally accessible to other registered clients).
5) Removing the end-of-hiding function to indicate to other asynchronous-hide clients that the hiding protocol has ended.
6) Adding/removing of a marker class with default name asynchronous-hide on the root element of the page.

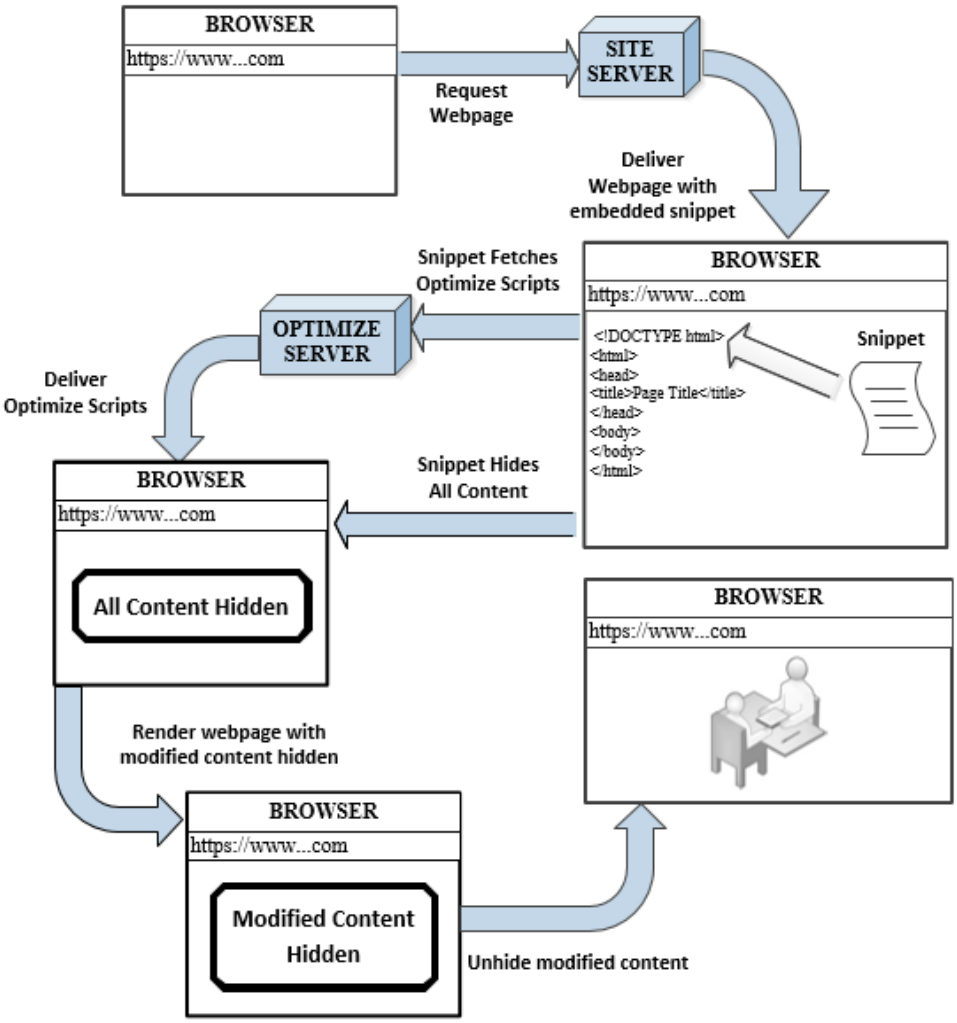
Asynchronous-hide clients may also participate at a later point, even when not listed in the asynchronous-hide snippet. In particular, a registered asynchronous-hide script can trigger the loading of other asynchronous-hide scripts and register them in the

asynchronous-hide protocol (i.e., add their identifier and state into the client state data structure) before it calls the endHide function. This approach, for example, may be used by online service providers that allow users to configure the asynchronous scripts they wish to add to their page, without modifying the actual page code.

The asynchronous-hide snippet also includes a small <style> element that defines the effects of the class, and sets the opacity of the root element to '0'. In some implementations, website owners may modify this style element to control the effect and extend of the hiding (i.e., to exclude specific elements). In some implementations, website owners may override the style element in their own cascading style sheets (CSS) files in order to customize it for different types of pages.

Example Operation

The diagram below shows the flow of the webpage hiding method for eliminating the flickering effect that occurs during asynchronous loading of A/B optimization scripts.



The method starts with a standard internet browser sending a webpage request to a website server storing webpages registered with the asynchronous-hide mechanism. In response to the request, the webserver sends the requested website, whose content includes an asynchronous-hide snippet, to the internet browser. In some implementations, the snippet may be embedded into the HEAD section of the webpage by using the "<script>" and "</script>" tags of the Hypertext Markup Language (HTML). During the rending of the webpage, the internet browser executes the embedded snippet, which triggers two actions. First, the snippet instructs the internet browser to fetch the flickering prevention scripts (optimization scripts) from the A/B testing tool's

optimization script server. Second, the snippet executes the web-hiding method to hide the display of the entire web page. For example, the snippet may configure a display attribute associated with the BODY section of the webpage such that the content of the BODY section is not displayed to the user of the web browser. In some implementations, the website owner may enclose the webpage content within a and element pair and control the SHOW/HIDE attribute of the element via Javascript.

The entire contents of the webpage remain hidden until the A/B testing server retrieves and delivers the optimization scripts to the browser. Since the contents of the webpage are fully hidden, a potential flickering event during this time will not reveal the contents of the original, unmodified webpage to the visitor. Advantageously, this improves both the visitor's webpage experience, and the capability of an A/B testing tool to accurately test the features of a webpage.

Before rending the webpage, the internet browser calls the "end-of-hiding" function of the snippet to un-hide the unmodified features of the webpage. For example, a web developer may create a second version of a webpage by changing the font color for all headlines of the original webpage from black to blue. Thus, the end-of-hiding function will reveal all features of the webpage except for the newly colored headlines. Instead, the internet browser displays the headlines as mere blanks. However, after the optimization scripts fully load and execute, the endHide function reverses the coding commands (e.g., HTML, java) that hid the modified features.

Product Help Page

In order to keep the default implementation compact, customization instructions and examples, as well as the endHide function may be listed in a convenient product help

page. This help page, together with the asynchronous-hide snippet, will cover the full extent of the webpage hiding method. The help page may provide instructions and examples on how to (1) modify the default maximum time limit until end-of-hide is called, (2) modify the default marker class to a name different than asynchronous-hide, (3) modify the default hiding effect and extend of that marker class, (4) explain how particular types of pages may override that effect and extend, (5) provide a completely different hide/unhide implementation, (6) add their own clients in the hide/unhide protocol, and (7) add a callback to be invoked at the time that end-of-hide is called and the page becomes visible.

Accordingly, this disclosure provides details regarding a webpage hiding method for eliminating the flickering effect during A/B testing through the use of asynchronous scripts without causing delay in the loading of a webpage.

Abstract

This document describes a technique for eliminating webpage flickering during A/B testing, also known as split or bucket testing, without delaying webpage load times. An asynchronous-hide snippet embedded into a webpage hides the whole webpage during the asynchronous loading of an A/B testing tools' optimization scripts. A snippet includes an end-of-hiding function to unhide the webpage, a timer to remove the function, and a globally accessible data structure to identify other scripts or clients. A separate endHide function invoked on start-up implements an un-hiding protocol, ensures snippet availability and script registration, marks the data structure to indicate the script hiding status, and removes the end-of-hiding function.